VISUAL BASIC

ศรันย์ ไมตรีเวช

# Visual BASIC 4.0 : สรุปความเปลี่ยนแปลง

***มาสำรวจดูกันว่า ไมโครซอฟต์ เปลี่ยนแปลงอะไรในวิชวลเบสิก 4.0 ให้เราบ้าง***

คงไม่เป็นการช้าเกินไปนะครับที่ผมจะนำความเปลี่ยนแปลงเกี่ยวกับอินเตอร์เฟสและรูปแบบภาษาของวิชวลเบสิกสรุปมาเสนอ เนื่องจากการเคลื่อนย้ายจากวิชวลเบสิก 3.0 มาเป็นเวอร์ชันล่าสุดของโปรแกรมเมอร์ส่วนใหญ่อยู่ในลักษณะ "ค่อย ๆ ทยอย" มากกว่าจะ "แห่" กันขึ้นมา

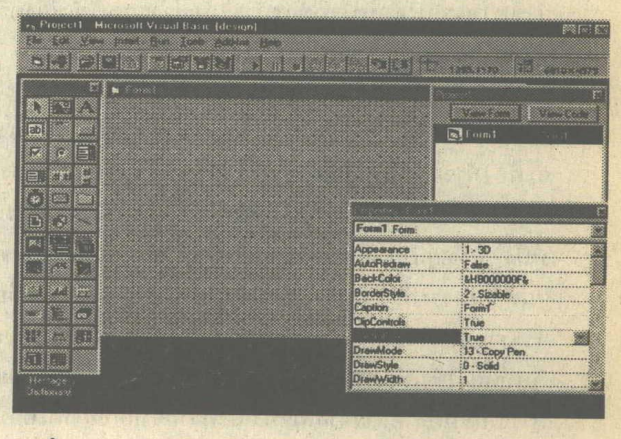
บทความเกี่ยวกับการสรุปความเปลี่ยนแปลงนี้คงเหมาะกับการเปรียบเทียบมากกว่า หากคุณไม่เคยเล่นวิชวลเบสิกเวอร์ชันก่อน ๆ ก็คงดู ๆ ไปว่ามีอะไรอยู่ในวิชวลเบสิก 4.0 บ้าง แต่หากคุ้นเคยกับวิชวลเบสิก 3.0 เป็นอันดี ก็คงได้เปรียบเทียบอย่างชัดเจนว่ามีอะไรเปลี่ยนแปลง มีอะไรเพิ่มขึ้น และมีอะไรที่ลดลง โดยผมจะเน้นการเปลี่ยนแปลงในส่วนของอินเตอร์เฟสและภาษาที่ถือเป็นหลักสำคัญสุดยอดในการพัฒนาแอพพลิเคชันด้วยวิชวลเบสิก

## ส่วนอินเตอร์เฟส

เมื่อมององค์รวมของสภาพแวดล้อมที่ใช้พัฒนาแอพพลิเคชันดังรูปที่ 1 ก็ดูเหมือน ๆ เดิมไม่ค่อยมีอะไรเปลี่ยนแปลงเท่าไหร่ แต่ความจริงเมื่อคุณเริ่มลงมือสร้างแอพพลิเคชัน ก็จะพบว่าต้องทำความเข้าใจกฎระเบียบและจดจำขั้นตอนลัดใหม่ ๆ ของวิชวลเบสิก 4.0 หลายอย่าง

แม้ได้มีการปรับปรุงแก้ไขเปลี่ยนแปลงหลายสิ่งหลายอย่างที่บอกแล้วว่าต้องลงมือสร้างแอพพลิเคชันถึงจะรู้ แต่สิ่งหนึ่งที่ได้เห็นกันชัด ๆ ต้องใช้ประจำ แล้วผิดหวังที่ไมโครซอฟต์ไม่เปลี่ยนแปลงให้เราคือ ทูลบ็อกซ์

การจัดวางรูปไอคอนอันเป็นอุปกรณ์สร้างคอนโทรลบนฟอร์มเป็นแบบมิติเดียวตามเดิม ซึ่งถือว่าผิดหลักเป็นอย่างยิ่งหากคำนึงถึงความมากมายก่ายกองของคอนโทรลพิเศษทั้งที่เป็นของไมโครซอฟต์เองและทั้งที่ซอฟต์แวร์เฮาส์ข้างนอกทำมาร่วมประกอบ ผิดหลักอย่างไร? ผิดตรงที่คุณต้องนั่งหากันตาเหล่แทบทุกครั้งกว่าจะเจอรูปไอคอนคอนโทรลที่ต้องการ เนื่องจากการเพิ่มเข้าหรือลดออกของรูปไอคอน ย่อมหมายถึงการเปลี่ยนตำแหน่งและแผนผังทั้งหมดของทูลบ็อกซ์ไปด้วย ผมเองแม้จะใช้วิชวลเบสิกหากินมาหลายปีก็ยังต้องเหลือบไปแลมาอยู่เสมอว่า Picture Box มันอยู่ที่ใด อ้าว



**รูปที่ 1 สภาพแวดล้อมเมื่อแรกเห็นของวิชวลเบสิก**

DirListBox เปลี่ยนไปอยู่ตรงไหนแล้ว ยิ่งเมื่อมีการโหลดคอนโทรลใหม่ ๆ ขึ้นมาใส่ทูลบ็อกซ์ยิ่งไปกันใหญ่ เพราะความไม่คุ้นรูปไอคอนหนึ่ง และความไม่มีตำแหน่งแน่ชัดของพวกมันอีกหนึ่ง

ลักษณะทูลบ็อกซ์ที่ดีควรมีการจัดแบ่งหมวดหมู่ไว้เป็นหน้า ๆ เช่น นี่คือกลุ่มที่เกี่ยวกับการแสดงหรือรับข้อมูลทั่วไป นี่คือกลุ่มที่เกี่ยวกับดาต้าเบสโดยเฉพาะ เมื่อต้องการสร้างคอนโทรลกลุ่มใดก็คลิ๊กเลือกจากแถบยื่นของหน้านั้น ๆ เชื่อว่าคุณที่เล่นเครื่องมือพัฒนาแอพพลิเคชันอื่นคงเคยผ่านตามากันบ้าง และย้อนกลับมามองว่าทูลบ็อกซ์มิติเดียวของวิชวลเบสิกเป็นเรื่องล้าสมัยอย่างยิ่ง

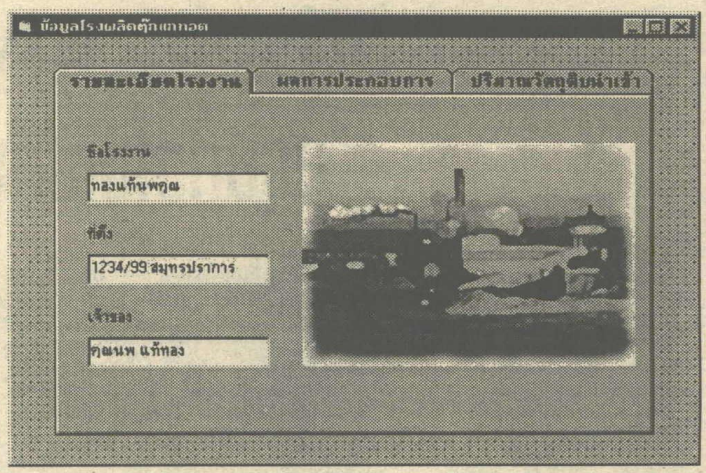
อย่างไรก็ตาม ข้อที่น่าติอันนี้ไม่ถือเป็นเรื่องร้ายแรงเท่าไหร่ เมื่อคิดเสียว่าวิชวลเบสิกมีคอนโทรลใหม่ดี ๆ ให้เราใช้อยู่เยอะ รวมทั้งระเบียบของอินเตอร์เฟสโดยรวมที่ "รีเอ็นจิเนียริ่ง" เสียใหม่ ไม่ว่าจะเป็นการปรับแต่งรูปลักษณ์ให้เข้าร่องเข้ารอยกันสนิทกับวินโดวส์ 95 กับการแก้ไขจุดบกพร่องที่เคยทำให้นักพัฒนาเกิดความสับสนงงงวยอยู่บ่อย ๆ เช่นแต่ก่อนเคยให้ใช้คำสั่ง Add File ทั้งเพื่อโหลดฟอร์มและเพื่อโหลดคอนโทรลได้ทั้งสองอย่าง ตอนนี้ก็แบ่งแยกกันอย่างชัดเจนว่าคำสั่งไหนจะให้โหลดอะไรกันแน่ หรืออย่างสิ่งที่ควรมีมาตั้งนานแต่เพิ่งมาโผล่เอาในเวอร์ชันนี้ เช่น ทูลทิป หรือป้ายประกาศชื่อตนเองของเหล่าไอคอนทั้งหลายเมื่อคุณนำเม้าส์พอยน์เตอร์ไปวางอยู่เหนือพวกมันประมาณวินาทีหนึ่ง

เป็นอันว่าคงรับกันได้นะครับ ขีดความพอใจในการทำงานกับวิชวลเบสิกน่าจะเป็นบวกแน่ ๆ เรามาพูดถึงทูลบ็อกซ์กันต่อ ที่คุณเห็นตามรูปที่ 1 นั้นคือทูลบ็อกซ์โดยดีฟอลต์ หมายความว่าเมื่อติดตั้งแล้วก็จะเห็นมันเป็นของมันอย่างนั้น สำหรับกลุ่มคอนโทรลมาตรฐานก็คงไม่ต้องมาแนะนำอะไรกัน เพราะรูปไอคอนเหมือนเดิม ตำแหน่งเดิมแค่เพิ่มสีสันนิดหน่อย เรามาดูว่าเฉพาะคอนโทรลหน้าตาแปลกใหม่กันดีกว่าครับว่ามีอะไรบ้าง และหน้าที่ในการติดต่อกับผู้ใช้ของมันเป็นอย่างไร

## SSTab

คอนโทรลตัวนี้ทำหน้าที่เป็น "สมุด" เล่มหนึ่งให้กับผู้ใช้ โดยผู้ใช้สามารถพลิกหน้าที่ต้องการได้อย่างรวดเร็วเพียงคลิ๊กแถบยื่น ซึ่งประกาศชื่อตนเองอยู่แล้วว่าเป็นหน้าเกี่ยวกับเรื่องอะไร เช่นในรูปที่ 2 ซึ่งผมยกตัวอย่างให้ดู

อันนี้เพื่อความเข้าใจและมองเห็นประโยชน์ของ SSTab ก็เห็นจะต้องท้าวความกันเสียหน่อย จำได้ไหมครับ สมัยที่เรายังไม่มีคอนโทรลตัวนี้ (หรือที่คล้ายกับคอนโทรลนี้) หากการแสดงข้อมูลในหมวดหมู่หนึ่ง ๆ มีอยู่มาก ต้องอาศัยคอนโทรลมาช่วยแสดงเยอะ เช่นข้อมูลโรงงานในตัวอย่างรูปที่ 2 นั้น เราต้องให้ผู้ใช้เขาเห็นทั้งรายละเอียดโรงงาน ภาพโรงงาน ตารางแสดงผลการประกอบการตลอดไตรมาส และจำนวนวัตถุดิบที่ต้องใช้ แน่นอนเราไม่ควรทำบนฟอร์มเพียงฟอร์มเดียว เพราะจะเป็นการเกลื่อนกลาดลายตาเกินไป และความจำกัดของพื้นที่ก็บังคับให้เราต้องบีบอัดรูปอินเตอร์เฟสให้เล็ก แสดงข้อมูลได้กระจิ๊ดเดียว หากผู้ใช้เขาอยากดูข้อมูลในลักษณะเปรียบเทียบเร็ว ๆ ก็จะเป็นการทรมานใจพวกเขาเป็นอันมาก




**รูปที่ 2 ลักษณะข้อมูลที่ถูกจัดแบ่งไว้ตามหมวดหมู่เป็นหน้า ๆ**

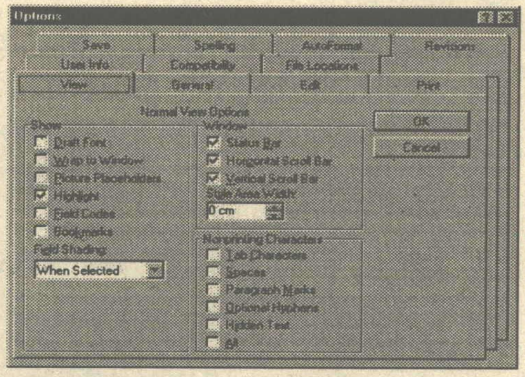
วิธีที่เราใช้ ๆ กันมาส่วนใหญ่ก็คือแบ่งการแสดงข้อมูลออกเป็นหลาย ๆ ฟอร์ม ซึ่งนับเป็นการสิ้นเปลืองทั้งทรัพยากร และทำให้ใจผู้ใช้ต้องกระโดดตามการเปลี่ยนฟอร์มไปมา ศูนย์รวมโฟกัสเคลื่อนได้ง่าย เวลาต้องเปลี่ยนสัก 5-6 ฟอร์มอย่างต่อเนื่องจะลืมไปเลยว่าหัวเรื่องใหญ่คืออะไร เพราะบางฟอร์มต้องใช้เวลาโหลดกันนาน แม้แต่ผู้บริหารที่มีความสามารถสูงในการจำแนกข้อมูล ซึ่งเมื่อจะค้นดูข้อมูลมักมีจุดประสงค์หลักที่แน่นอนเป็นตัวนำ ก็อาจขาดการจับต้นชนปลายได้ติดตลอด เหตุก็คือการเสนอข้อมูลไม่จับให้ใจจดเท่าที่ควร

การรวมข้อมูลหมวดเดียวกันไว้ในหนึ่งฟอร์ม จะโฟกัสของความสนใจที่ปักอยู่กับกรอบแสดงเพียงกรอบเดียว การคลิ๊กเลือกและรอโหลดฟอร์มหมดไป ทำให้ไม่เขวจากหัวเรื่องใหญ่ง่าย ๆ วิธีการก็คือทำคอนโทรลออกมาตัวหนึ่ง มีชั้นการแสดงข้อมูลหลายมิติ เหมือนสมุดโทรศัพท์ แบ่งหมวดหมู่ด้วยแถบยื่นเป็นตัวอักษรให้คุณใช้ปลายนิ้วเหนี่ยวเปิดตามต้องการได้โดยง่าย

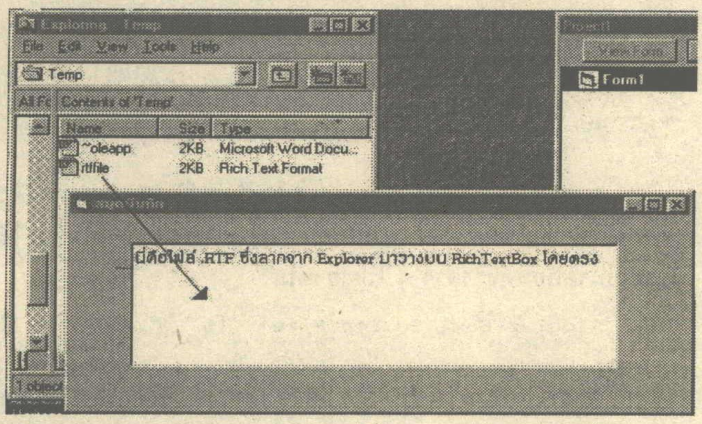
SSTab วิเศษตรงที่แต่ละแผ่นแต่ละหน้าแสดงข้อมูลนั้นสามารถบรรจุด้วยคอนโทรลอะไรก็ได้ และเป็นจำนวนมากได้เท่าที่คุณต้องการ (หากมากเกินไปผู้ใช้เขาก็จะดูไม่รู้เรื่องเหมือนกันนะครับ) อย่างเช่นหน้าแรกตามตัวอย่างรูปที่ 2 นี้ ผมใส่เข้าไปสามชนิด คือ Picture Box, Text Box และ Label แล้วซ่อนตารางกับคอนโทรลอื่นไว้ในแผ่นที่ 2 และ 3 ซึ่งบอกผู้ใช้ไว้ว่าเป็นหน้าแสดงข้อมูลเกี่ยวกับผลการประกอบการและปริมาณวัตถุนำเข้า

ทำได้ขนาดนี้มีอยู่เพียงไม่กี่ขั้นตอนเองครับ พอสร้างคอนโทรลขึ้นบนฟอร์มแล้วกำหนดพรอพเพอร์ตี้ส์แค่ 2-3 ตัว ดังนี้

- Caption: ใช้กำหนดว่าจะให้แถบยื่นมันประกาศอะไร คุณต้องคลิ๊กเลือกแถบยื่นที่ต้องการเสียก่อนนะครับ




**รูปที่ 3 หมวดหมู่จำนวนมากที่บรรทัดเดียว รับไม่ไหว**




**รูปที่ 4 คุณสมบัติอันหนึ่งของ RichTextBox คือการรับไฟล์นามสกุล .RTF ได้โดยตรงจาก Explorer ด้วยวิธีลากวาง**

- Tabs: เป็นจำนวนหน้าการแสดงที่ต้องการ โดยดีฟอลต์มีอยู่สาม ซึ่งเผอิญ (อย่างจงใจ) พอดีกับหมวดหมู่ข้อมูลที่ผมต้องการแสดงในที่นี้ ผมเลยไม่ต้องเข้าไปยุ่งเกี่ยวอะไรกับมัน

- TabsPerRow: เป็นตัวช่วยแบ่งระนาบซ้อนกลุ่มแสดงข้อมูลเข้าไปอีก เนื่องจากจำนวนหมวดหมู่บางครั้งซอยได้เป็นสิบ เช่นหน้า Options ของไมโครซอฟต์เวิร์ดดังรูปที่ 3

พรอพเพอร์ตี้ส์ TabsPerRow จะให้คุณกำหนดตามความเหมาะสมเฉพาะหน้าว่าระนาบหนึ่ง ๆ จะให้แสดงกี่แผ่น เช่นในหน้าต่าง Options ของไมโครซอฟต์นี้ หากมองว่าทำด้วย SSTab ก็จะมีค่าเป็น 4 (สังเกตด้วยนะครับว่าจำนวนหมวดหมู่ที่มีอยู่จริง ๆ คือ 11 ซึ่งหาร 4 ไม่ลงตัว ในระนาบที่สองจึงปัดเศษแสดงเพียงสามให้โดยอัตโนมัติ

## RichTextBox

คอนโทรลตัวนี้เอาไว้ให้ผู้ใช้เขาบันทึกอะไรตามเรื่องตามราว อาจเป็นหมายเหตุ ข้อสังเกต หรือความเห็นส่วนตัว ซึ่งมีกับข้อมูลหนึ่ง ๆ คุณจะนึกเปรียบเทียบกับ TextBox ก็ได้ แต่มันห่างชั้นกันราวฟ้ากับดิน เพราะสำหรับ TextBox นั้น หากเป็นคนผมก็รู้สึกว่าเจ้าหมอนี่ซื่อบื้อและยากจนยังไงบอกไม่ถูก ส่วน RichTextBox นั้นรวยสมชื่อ ฉลาดสมตัวเลยครับ ทั้งคุณและผู้ใช้เล่นอะไรกับมันได้หลายอย่าง แม้กระทั่งเปลี่ยนขนาดฟอนต์ให้กับแต่ละบรรทัด

การใช้ RichTextBox ให้ได้ประโยชน์สูงสุดนั้น คุณรู้จักแค่พรอพเพอร์ตี้ส์สำคัญไม่ได้ ต้องเข้าใจครอบคลุมให้หมด ทั้งเมธอดและอีเวนต์ รวมทั้งขั้นตอนลัดอันสะดวกและน่าทึ่งสำหรับผู้ใช้ เช่น การลากไฟล์นามสกุล .RTF ซึ่งเป็นฟอร์แมตแบบ Rich Text มาวางบน RichTextBox ขณะที่โปรแกรมกำลังรัน ผลที่ได้คือข้อความในไฟล์ดังกล่าวจะปรากฏให้เห็นใน RichTextBox ทันทีดังรูปที่ 4

ในการใช้งานจริง ผู้ใช้เขาคงรู้สึกไม่สะดวกนักหากต้องทำงานควบคู่กับ Explorer คุณก็สามารถใช้ FileListBox มาช่วยได้ แต่ต้องเขียนโค้ดกันหน่อย ไม่ใช่ของสำเร็จรูปเหมือนอย่างที่ทำกันในรูปที่ 4

อีกเรื่องที่ต้องหมายเหตุไว้ก่อนแนะนำให้รู้จักกับการใช้งานหลักนั่นคือคุณต้องนำไฟล์ RICHTX32.OCX ไปติดตั้งใน \windows\system ของเครื่องผู้ใช้ด้วยนะครับ ไม่อย่างนั้นเขาจะใช้ RichTextBox ไม่ได้ อันนี้สำคัญมาก หากคุณเพิ่งใช้วิชวลเบสิกอาจเผลอนึกว่าเห็น RichTextBox ปรากฏอยู่ในทูลบ็อกซ์อยู่แล้วโดยดีฟอลต์ ก็คงหมายถึงว่ามันเป็นคอนโทรลสร้างสำเร็จรูปในระดับเอ็นจิน พร้อมจะเป็นส่วนหนึ่งส่วนเดียวกับแอพพลิเคชันทันที แต่ความจริงไม่ใช่ มันเป็นคัสตอมคอนโทรลซึ่งต้องอิงอาศัยไฟล์เฉพาะมาสร้างในตอนรันอีกทีหนึ่ง

คราวนี้มาดูประโยชน์หลักทั่วไปของ RichTextBox กันครับ ผมจะคัดเฉพาะพรอพเพอร์ตี้ส์และเมธอดที่เป็นตัวประโยชน์ใช้สอยเด่น ๆ ของมันมาเสนอไปก่อน เพราะถ้าขุดมาทั้งหมดก็คงยืดยาวเกินไป

## พรอพเพอร์ตี้ส์หลัก

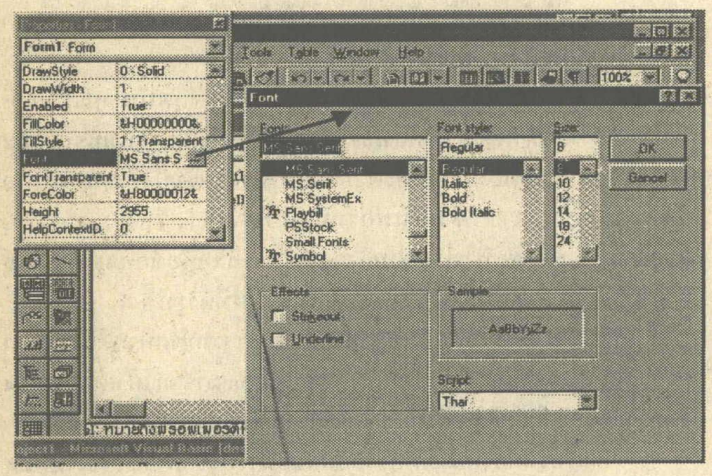
1. Font: เมื่อตอนอยู่ในเวอร์ชัน 3.0 คุณคงจำได้ดีว่ากลุ่มพรอพเพอร์ตี้ส์ที่ขึ้นด้วย Font นั้นมีอยู่เยอะมาก และทำให้ลิสต์ของพรอพเพอร์ตี้ส์มันยืดยาวกว่าที่ควรจะเป็นตั้งเยอะ เช่น FontSize, FontName ฯลฯ

ในเวอร์ชัน 4.0 นี้เข้าท่ากว่าเดิมเยอะ คือพรอพเพอร์ตี้ส์ Font กลายเป็นออบเจ็กต์ตัวหนึ่ง มีพรอพเพอร์ตี้ส์แยกย่อยได้อีก และเพราะอย่างนี้ เมื่ออยู่ในโหมดออกแบบ คุณจึงเห็นหน้าต่างพรอพเพอร์ตี้ส์ในบรรทัด Font สามารถเรียกหน้าต่างกำหนดคุณสมบัติของฟอนต์โดยเฉพาะมาให้ได้ดังรูปที่ 5

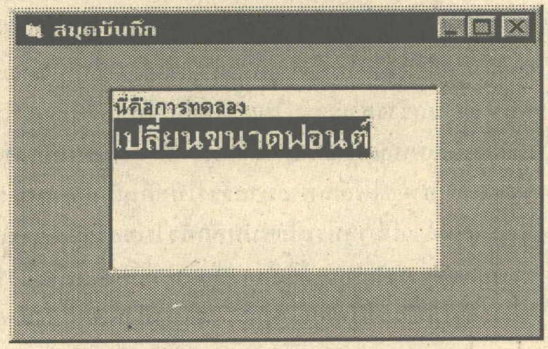
และในขณะรันนั้น คุณก็สามารถเขียนโค้ดกำหนดโครงสร้างความเป็นฟอนต์ด้วยรูปแบบที่ง่ายขึ้น ย้อนกลับมาดูได้เข้าใจกว่าเดิม เช่น

```
RichTextBox1.Font.Underline =
Not  RichTextBox1.Font.Underline
```

หรือถ้ามีพรอพเพอร์ตี้ส์หลายตัวต้องเปลี่ยนก็ใช้โครงสร้าง With...End With ซึ่งทำให้คุณสามารถเปลี่ยนพรอพเพอร์ตี้ส์ของออบเจ็กต์หนึ่ง ๆ ได้หลายตัวโดยอ้างชื่อออบเจ็กต์เพียงหนเดียว ทำให้เกิดกรอบที่อ่านทำความเข้าใจง่ายโดยเฉพาะเมื่อต้องเปลี่ยนพรอพเพอร์ตี้ส์จำนวนมาก ๆ



รูปที่ 5 การเรียกหน้าต่างกำหนดคุณสมบัติของฟอนต์ด้วยปุ่มสามจุด




รูปที่ 6 การขยายขนาดฟอนต์ด้วยการคลิ๊กฟอร์ม

```
With RichTextBox1.Font
      .Size = 20
      .Name = "BrowalliaUPC"
      .Bold = True
End With
```

ต้องทำความเข้าใจไว้นะครับว่า การกำหนดค่าให้กับ Font ของ RichTextBox นี้ เป็นการกำหนดทีเดียวได้ผลทั่วไปทั้งหมด หมายความว่าอักขระที่เห็นใน RichTextBox ทุกตัวจะตกอยู่ภายใต้การกำหนดทั้งหมด ซึ่งในการใช้จริง บางครั้งผู้ใช้เขาไม่ต้องการอย่างนั้น เขาต้องการแค่เพียงบางส่วน อันนี้เราก็ทำได้ด้วยพรอพเพอร์ตี้ส์ตัวต่อไปครับ

2. กลุ่ม Sel: หมายถึงพรอพเพอร์ตี้ส์ที่ขึ้นต้นด้วย Sel (ย่อมาจาก Selected) เช่น SelFontSize, SelFontName ฯลฯ ซึ่งจะทำให้โปรแกรมเมอร์สามารถเปลี่ยนคุณสมบัติของฟอนต์เฉพาะกลุ่มได้ขณะรัน ไม่ใช่เปลี่ยนทั้งหมดทั่ว RichTextBox เหมือนเมื่อตอนใช้พรอพเพอร์ตี้ส์ Font

เงื่อนไขคือต้องมีการโฟกัส (มีแถบมืดคาด) กลุ่มข้อความส่วนใดส่วนหนึ่งเสียก่อน กลุ่มข้อความที่ถูกโฟกัสนั้นแหละครับที่เรากำหนดเปลี่ยนคุณสมบัติได้ด้วยพรอพเพอร์ตี้ส์กลุ่ม Sel นี้ ยกตัวอย่างเช่น เราต้องการให้ผู้ใช้มีสิทธิ์ขยายหรือลดขนาดฟอนต์ได้ทีละสองเท่า เมื่อเขาโฟกัสข้อความแล้วคลิ๊กบนที่ว่างบนฟอร์ม ถ้าคลิ๊กด้วยปุ่มซ้ายหมายถึงเพิ่มขนาด ถ้าคลิ๊กด้วยปุ่มขวาก็หมายถึงลดขนาด เช่นตัวอย่างดังรูปที่ 6 นั้น เดิมขนาดฟอนต์เท่ากันทั้งสองบรรทัด แต่พอผู้ใช้โฟกัสข้อความบรรทัดที่สองแล้วคลิ๊กที่ว่างของฟอร์มด้วยปุ่มซ้ายของเม้าส์ ก็ปรากฏว่าขนาดเพิ่มขึ้นเป็นสองเท่าทันที

ที่เราทำได้อย่างนั้นก็ต้องเขียนโค้ดกันอย่างนี้ครับ

```
Private Sub Form_MouseDown(Button As Integer,
Shift As Integer, X As Single, Y As Single)
   If Button = 1 Then
      RichTextBox1.SelFontSize =
RichTextBox1.SelFontSize * 2
   ElseIf Button = 2 Then
      RichTextBox1.SelFontSize =
RichTextBox1.SelFontSize / 2
   End If
End Sub
```

หากคุณไม่ต้องการกำหนดขนาดฟอนต์ในลักษณะอ้างอิงเปรียบเทียบกับของเดิม ต้องการใส่ตัวเลขลงไปตรง ๆ ก็ได้ครับ ใช้ตัวเลขขนาดฟอนต์จริงตรง ๆ ไปเลย เช่น

```
RichTextBox1.SelFontSize = 16
```

พรอพเพอร์ตี้ส์ในกลุ่ม Sel อื่น ๆ นั้นคงต้องขอรบกวนให้คุณไปศึกษาและทดลองใช้กันเอง แต่ละตัวใช้ง่ายแล้วก็ตรงตามลักษณะวิธีที่คุณเคยผ่านมาในวินเวิร์ดทั้งนั้นแหละครับ มีข้อสังเกตอยู่อย่างหนึ่งว่าคุณจะใช้อีเวนต์ไหนของคอนโทรลใดมาเป็นตัวกำหนดการเปลี่ยนแปลงคุณสมบัติของฟอนต์ที่ถูกโฟกัสก็ได้ ยกเว้นอีเวนต์ของ RichTextBox เอง เนื่องจากเมื่อเกิดการกระทำอย่างใดอย่างหนึ่งกับตัวของ RichTextBox ก็จะทำให้ไฮไลท์ของกลุ่มข้อความเสียไปทันที

3. FileName: เป็นพรอพเพอร์ตี้ส์ที่ช่วยให้คุณกำหนดไฟล์ที่จะนำมาแสดงด้วย RichTextBox ได้โดยตรง โดยฟอร์แมตของไฟล์ไม่จำกัดนะครับว่าต้องเป็นเฉพาะ .RTF เท่านั้น จะเป็นแอสกีก็ได้ (มักใส่นามสกุล .TXT กันมากกว่าอย่างอื่น) เช่นถ้าคุณต้องการให้ RichTextBox มันแสดงข้อความในไฟล์ชื่อ TextFile.txt ตั้งแต่โหลด ก็เขียนได้ว่า

```
Private Sub Form_Load()
   RichTextBox1.FileName = "\Temp\TextFile.txt"
End Sub
```

มีข้อแม้อยู่นิดหนึ่งว่าพรอพเพอร์ตี้ส์นี้ใช้ได้ตอนอยู่บนวินโดวส์ 95 หรือวินโดวส์เอ็นทีเท่านั้นนะครับ ด้วยเหตุผลเกี่ยวกับการจัดการไฟล์ข้างในของ RichTextBox หากคุณใช้วิชวลเบสิก 4.0 เวอร์ชันบนวินโดวส์ 3.x ก็หมดสิทธิ์เล่นพรอพเพอร์ตี้ส์นี้

4. DataSource และ DataField: เป็นพรอพเพอร์ตี้ส์ที่แสดงให้เห็นว่า RichTextBox สามารถเป็นบาวนด์คอนโทรลได้ วิธีใช้คือระบุชื่อ Data Control ด้วย DataSource และระบุชื่อฟิลด์ (ซึ่งเราเลือกเทเบิลไว้ใน Data Control แล้ว) ด้วย DataField

## เมธอดหลัก

1. LoadFile: ทำหน้าที่โหลดเอาไฟล์ .RTF หรือ .TXT (ไม่จำเป็นต้องนามสกุลตรงตัวแบบนี้ ขอให้ฟอร์แมตใช่เป็นใช้ได้) ขึ้นมาแสดงด้วย RichTextBox มีซินแท็กซ์คือ

```
ชื่อคอนโทรล.LoadFile ชื่อไฟล์พร้อมพาธ, ฟอร์แมตที่ต้องการ
```

สำหรับอาร์กิวเมนต์ตัวที่สองคือฟอร์แมตของไฟล์ที่ต้องการให้โหลดมานั้น เป็นไปได้สองอย่างคือ rtfRTF (ต้องการโหลดฟอร์แมต .RTF) หรือ rtfTEXT (ต้องการโหลดฟอร์แมตแอสกี) ยกตัวอย่างเช่น

```
RichTextBox1.LoadFile "\Temp\TextFile.txt",
rtfTEXT
```

2. SaveFile: ทำหน้าที่จัดเก็บข้อความทั้งหมดที่กำลังอยู่ใน RichTextBox ลงเป็นไฟล์ฟอร์แมต .RTF หรือ .TXT (คุณอาจตั้งชื่อนามสกุลเป็นอย่างอื่นก็ได้) มีซินแท็กซ์คือ

```
ชื่อคอนโทรล.SaveFile(ชื่อไฟล์พร้อมพาธ, ฟอร์แมตที่ต้องการ)
```

สังเกตให้ดีนะครับว่า SaveFile รับชื่อกับฟอร์แมตในลักษณะพารามิเตอร์ของฟังก์ชัน ต้องมีวงเล็บด้วย ยกตัวอย่างเช่น

```
RichTextBox1.SaveFile("\Temp\TextFile.txt",
rtfTEXT)
```

3. Find: เมธอดตัวนี้มีประโยชน์สูงมาก เพราะใช้ในการค้นหาคำที่ต้องการ หากผู้ใช้กำลังเกี่ยวข้องกับไฟล์ใหญ่ ๆ ก็น่าจะเห็นค่าของคุณสมบัติค้นหาคำได้เป็นพิเศษ ซินแท็กซ์ของ Find คือ

```
ชื่อคอนโทรล.Find(คำที่ต้องการค้นหา, ตำแหน่งเริ่ม, ตำแหน่งสุดท้าย, วิธีค้น)
```

สำหรับตำแหน่งเริ่มนั้น หมายถึงตัวเลขตำแหน่งอักขระที่คุณระบุได้ว่าจะให้เริ่มที่ตรงไหน เช่นถ้ากำหนดเป็น 100 ก็หมายความว่า RichTextBox จะเริ่มค้นหาคำในตัวมันนับตั้งแต่ตำแหน่งอักขระที่ 100 เป็นต้นไป ส่วนตำแหน่งสุดท้ายก็เป็นตรงข้าม คือถ้าไปถึงตำแหน่งที่คุณระบุแล้วยังไม่เจอ มันก็จะไม่ค้นหาต่อ จะจบการค้นหาและถือว่าไม่พบคำที่ต้องการไปเลย สำหรับตำแหน่งเริ่มและตำแหน่งสุดท้ายนี้ คุณจะกำหนดค่าหรือไม่ก็ได้ ถ้าหากละไว้ก็ต้องใส่เครื่องหมาย , แทนที่ลงไปด้วย เนื่องจากการรับอาร์กิวเมนต์จะถือตำแหน่งเป็นสำคัญ (หากคุณกระโดดข้ามโดยไม่ใส่อะไรเลย เช่น คำที่ต้องการค้น ตำแหน่งสุดท้าย อย่างนี้ค่าตำแหน่งสุดท้ายที่คุณตั้งใจกำหนดก็จะกลายเป็นตำแหน่งเริ่มไป

สำหรับอาร์กิวเมนต์ที่เป็นวิธีค้นนั้นทำได้สามแบบหรือผสมกันทั้งสามแบบ คือ

1. rtfWholeWord (หรือใช้ค่าตัวเลข 2) มีไว้เพื่อค้นหาจากทั้งคำเทียบเท่าที่คุณต้องการ เช่นถ้ามีคำว่า ringing อยู่ใน RichTextBox แต่คุณต้องการค้นหาเฉพาะคำว่า ring แบบนี้ มันจะหาไม่เจอในแบบวิธีค้น rtfWholeWord อันนี้สำคัญมากนะครับ **อย่าใช้วิธีค้นแบบ rtfWholeWord กับข้อความภาษาไทยเป็นอันขาด** เพราะภาษาไทยเราไม่มีเว้นวรรค นอกเสียจากจะจบประโยคขึ้นประโยคใหม่ RichTextBox ไม่มีความสามารถในการแบ่งคำด้วยตรรกะหรือดิกชันนารี ดังนั้นมันจะถือว่าประโยคไทยทั้งประโยคเป็นคำเพียงคำเดียว

2. rtfMatchCase (หรือใช้ค่าตัวเลข 4) กำหนดให้การค้นหาคำเป็นไปตามแบบอักขระเล็กใหญ่ เช่นถ้าคำที่คุณต้องการหาคือ Big ปกติถ้าไม่อยู่ในแบบวิธีเลือก rtfMatchCase แล้วเมื่อ RichTextBox เจอคำว่า big มันก็จะถือว่าใช่ด้วย แต่หากคุณเลือก rtfMatchCase แล้วล่ะก็ คราวนี้ RichTextBox จะคัดเฉพาะคำว่า Big มาให้เท่านั้นครับ

3. rtfNoHighlight (หรือใช้ค่าตัวเลข 8) เมื่อ RichTextBox ค้นเจอคำที่คุณต้องการ มันจะไม่คาดแถบมืดให้กับคำนั้น

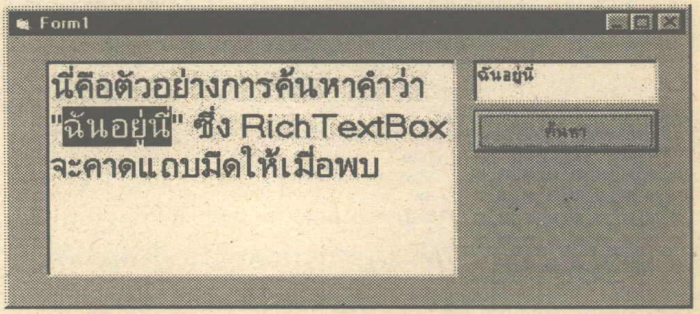
หากคุณต้องการให้แบบวิธีค้นผสมกัน ก็เพียงนำตัวเลขซึ่งเป็นค่าแทนแบบค้นหามาบวกกัน เช่นต้องการเลือกค้นแบบ rtfWholeWord กับ rtfMatchCase ก็เอา 2 มารวมกับ 4 เป็น 6

สำหรับเมธอด Find นี้ต้องมีตัวแปรมารับนะครับ แล้วก็เกี่ยวข้องกับพรอพเพอร์ตี้ส์ HideSelection ของ RichTextBox ด้วย กล่าวคือ HideSelection ต้องเป็น False เท่านั้น เมื่อมีการค้นพบคำที่คุณต้องการผู้ใช้ถึงจะเห็นแถบมืดคาด หากเป็น True ก็จะไม่เห็น ซึ่งผมประหลาดใจเอามาทีเดียวที่โดยดีฟอลต์แล้ว แทนที่ HideSelection จะเป็น False ตามควร ดันกลายเป็น True ไปได้ ถ้าเจอหน้าคนออกแบบคงต้องจับเข่าคุยกันหน่อยล่ะครับว่ามีเหตุผลอะไรถึงทำให้พวกเราต้องลำบากลำบนไปเซต HideSelection กันอีก เพราะคุณสมบัติในการค้นหาคำเป็นสิ่งปกติที่ทุกคนต้องการอยู่แล้ว

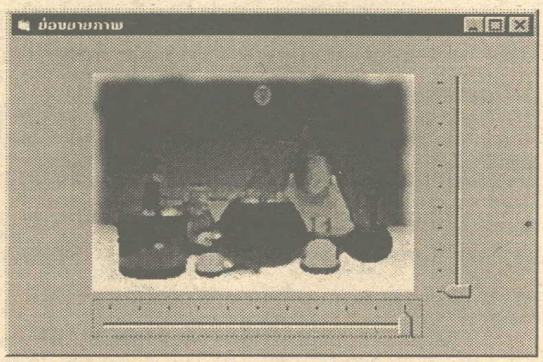
ตัวอย่างต่อไปนี้คือการกำหนดวิธีค้นหาที่เป็นไปได้

```
x = RichTextBox1.Find(Text1.Text)
x = RichTextBox1.Find("ฉันอยู่นี่", 100, 200,)
x = RichTextBox1.Find(Label1.Caption, , ,
    rtfMatchCase)
x = RichTextBox1.Find("String  found", , , 6)
```

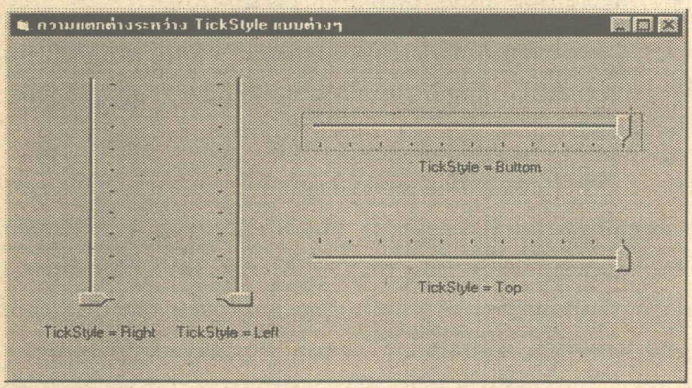
และตัวอย่างของหน้าตาฟอร์มที่เราใช้ค้นหาคำใน RichTextBox ก็เช่นรูปที่ 7




**รูปที่ 7 การใช้ TextBox เป็นผู้รับคำที่ผู้ใช้ต้องการค้นหาใน RichTextBox**




**รูปที่ 8 การวาง Slider ทั้งสองแนว เพื่อย่อ-ขยายภาพ ได้ทั้งตามแนวนอน และแนวตั้ง**




**รูปที่ 9 ความแตกต่างระหว่างการวางตำแหน่งของขีดจากการกำหนดด้วยพรอพเพอร์ตี้ส์ TickStyle**

หากคุณต้องการบอกอะไรกับผู้ใช้เป็นพิเศษ เช่น เมื่อค้นไม่พบก็รายงานว่าค้นไม่พบ ทางที่เป็นไปได้คือใช้ค่าอินเด็กซ์ที่เมธอด Find จะคืนมาให้กับตัวแปร x ค่าอินเด็กซ์นี้จะเป็นตำแหน่งอักขระแรกของคำที่มันค้นพบ หากค้นไม่พบคำใดเลยค่าที่คืนมาจะเป็น -1 ซึ่งคุณก็กำหนดเงื่อนไขได้ครับว่าหาก x เป็น -1 จะให้มีเมสเสจบ็อกซ์รายงานผู้ใช้ว่าอย่างไร

## Slider

คอนโทรลตัวนี้ทำหน้าที่เป็น "ตัวเลื่อน" ให้สมชื่อนั่นแหละครับ เทียบได้กับ ScrollBar แต่ดีกว่ากันในแง่ของความเห็นได้ชัด ทำให้ผู้ใช้เกิดความรู้สึกว่าเล่นง่าย มีขีดบอกอย่างแน่นอนว่ากำลังอยู่ตำแหน่งใดของช่วงความกว้างที่เลือกได้ทั้งหมด โดยทั่วไปเราจะวาง Slider ตามแนวนอน ซึ่งก็เป็นค่าดีฟอลต์อยู่แล้ว แต่อาจมีบางกรณีที่ควรวางตามแนวตั้ง เช่นเมื่อเราต้องการใช้มันลดหรือขยายขนาดของคอนโทรล Image ตามแนวตั้งเป็นต้น ทั้งแบบวางแนวนอนและแนวตั้งผมได้แสดงให้ดูในรูปที่ 8 ซึ่งเป็นตัวอย่างที่เรานำ Slider ไปประยุกต์ใช้กับแอพพลิเคชันที่มีการปรับขนาดของคอนโทรล Image

มาดูกันครับว่าพรอพเพอร์ตี้ส์และเมธอดหลักของ Slider มีอะไรบ้าง

### พรอพเพอร์ตี้ส์หลัก

1. Min: เป็นค่าของขีดต่ำสุด โดยดีฟอลต์จะเป็น 0
2. Max: เป็นค่าของขีดสูงสุด โดยดีฟอลต์จะเป็น 10
3. Value: เป็นค่าของขีดปัจจุบันที่เห็นเข็มชี้อยู่
4. Orientation: กำหนดให้ Slider วางตามแนวนอน (Horizontal "หรือค่า = 0") หรือแนวตั้ง (Vertical "หรือค่า = 1")
5. TickStyle: กำหนดว่าจะให้ขีดวางอยู่ซ้าย/ขวา/บน/ล่าง ของเข็ม เช่น เมื่อคุณกำหนดให้ Slider ถูกวางตามแนวตั้ง จะเห็นว่าโดยดีฟอลต์ขีดจะอยู่ทางด้านขวา (ซึ่งก็ทำให้เข็มชี้ทางขวาตามไปด้วย) ซึ่งในทางปฏิบัติจริงอาจไม่เข้ากับการใช้งาน เราอาจต้องการให้ขีดอยู่ทางด้านซ้ายของเข็มมากกว่า เป็นไปตามการจัดวางเฉพาะกิจ ดังที่เห็นในรูปตัวอย่างที่ 8 นั้น ผมก็ต้องการให้เข็มชี้มาทางด้านซ้าย ซึ่งเป็นตำแหน่งที่วางตามธรรมชาติของรูป

เพื่อให้คุณเห็นภาพเปรียบเทียบทั้งหมด ผมได้แสดงให้เห็นข้อแตกต่างทั้งตามแนวนอนและแนวตั้งในรูปที่ 9

ความจริงพรอพเพอร์ตี้ส์ TickStyle มีค่าทั้งหมดที่เป็นไปได้อยู่ 4 ค่า คือ

1. Bottom/Right (0) กำหนดให้ขีดอยู่ด้านล่างถ้าเป็นแนวนอน และอยู่ด้านขวาถ้าเป็นแนวตั้ง
2. Top/Left (1) กำหนดให้ขีดอยู่ด้านบนถ้าเป็นแนวนอน และอยู่ด้านซ้ายถ้าเป็นแนวตั้ง
3. Both (2) กำหนดให้ขีดปรากฏทั้งสองด้าน ไม่ว่าจะเป็นซ้าย ขวา บนหรือล่าง อันนี้คุณต้องขยายขนาดของคอนโทรลให้ใหญ่พอด้วยนะครับถึงจะเห็น เพราะขนาดดีฟอลต์จะแสดงได้เพียงด้านเดียว นอกจากขีดปรากฏทั้งสองด้านแล้ว คุณจะเห็นเข็มมันเปลี่ยนเป็นปุ่มสี่เหลี่ยมแทนรูปเข็มชี้ด้วย
4. No Ticks (3) กำหนดให้ขีดหายไปทั้งหมด อย่างไรก็ตามคุณจะเห็นเข็มยังคงอยู่ ซึ่งคราวนี้จะเป็นค่าดีฟอลต์ซึ่งกำหนดไม่ได้ นั่นคือถ้าเป็นแนวตั้งจะเห็นชี้ไปทางขวา ถ้าเป็นแนวนอนจะเห็นชี้ไปข้างบน

### เมธอดหลัก

GetNumTicks: คืนค่ามาเป็นจำนวนขีด ใช้ประโยชน์ตอนคุณต้องการทราบจำนวนขีดซึ่งอาจเปลี่ยนแปลงได้ตลอดเวลา โดยคุณไม่ทราบแน่นอนว่ามีประมาณเท่าไหร่ เช่น คุณอาจเซตพรอพเพอร์ตี้ส์ Max ให้แปรผันไปตามจำนวนวันของแต่ละเดือน ซึ่งเป็นไปได้คือ 28, 29, 30 และ 31 ถ้าหากใช้สายตาประมาณเอาจะไม่มีทางทราบได้เลยว่าตอนนี้มีจำนวนขีดอยู่เท่าไหร่แน่ เนื่องจากมันใกล้เคียงกันและไม่อยู่ในวิสัยที่สายตาธรรมดาจะกะเกณฑ์เอาเอง

ขอให้จำด้วยนะครับว่าเมื่อจำนวนขีดนั้น จะมีค่าเท่ากับ Max-Min+1 เสมอ อย่างเช่นจำนวนขีดโดยดีฟอลต์นี้ จะมีค่าเท่ากับ 10-0+1 = 11 ถ้าหากคุณลองใช้เมธอด GetNumTicks ดูก็จะเห็นว่า Slider รายงานค่าออกมาเป็น 11

ในการใช้ Slider นั้น ผู้ใช้ไม่จำเป็นต้องใช้เม้าส์ในการเลื่อนเข็มหรอกนะครับ เขาใช้ปุ่มลูกศรทั้งสี่ของคีย์บอร์ดก็ได้ โดยที่ปุ่มซ้ายกับปุ่มล่างจะทำให้เข็มเลื่อนมาทางซ้ายมือ ส่วนปุ่มขวากับปุ่มบนจะทำให้เข็มเลื่อนมาทางขวามือ อันนี้เป็นสิ่งที่คุณสมควรบอกกับผู้ใช้ให้ชัดเจนด้วย เพราะการทำงานจริงกับ Slider นั้น หลายคนถนัดใช้คีย์บอร์ดมากกว่าเม้าส์ โดยเฉพาะในงานที่ต้องการเลื่อนเข็มไปเพียงทีละหนึ่งหรือสองขีด ต่อไปนี้เป็นตัวอย่างโปรแกรมเพื่อดักเหตุการณ์จากปุ่มศรทั้งสี่ โดยเราจะให้โฟกัสเปลี่ยนจากคอนโทรลปัจจุบันไปยัง Slider แล้วเลื่อนเข็มให้ขีดหนึ่งตามปุ่มที่เขากดด้วย การเปลี่ยนโฟกัสจะได้รับการยกเว้นเมื่อคอนโทรลปัจจุบันเป็น TextBox (ซึ่งผู้ใช้เขาอาจเจตนาเลื่อนเคอร์เซอร์ภายใน TextBox นั้นเอง)

```
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
    If TypeOf Me.ActiveControl Is TextBox Then Exit Sub
    Slider1.SetFocus

    If KeyCode = 37 Or KeyCode = 40 Then 'ผู้ใช้กดปุ่มศรซ้ายหรือปุ่มล่าง
        Slider1.Value = Slider1.Value - 1
    ElseIf KeyCode = 39 Or KeyCode = 38 Then 'ผู้ใช้กดปุ่มศรขวาหรือปุ่มบน
        Slider1.Value = Slider1.Value + 1
    End If
End Sub
```

อย่าลืมเซตพรอพเพอร์ตี้ส์ KeyPreview ของฟอร์มให้เป็น True ด้วยนะครับ เหตุการณ์จากคีย์บอร์ดจะได้เข้าไปที่ฟอร์มก่อนคอนโทรลปัจจุบัน แล้วอย่าใช้อีเวนต์ KeyPress เป็นตัวดัก เพราะเหตุการณ์จากปุ่มศรทั้งสี่จะไม่เข้าไปในอีเวนต์ KeyPress

แล้วก็ต้องหมายเหตุไว้อีกข้อหนึ่ง คือจะมีคอนโทรลบางชนิด เช่น Option จะมีอภิสิทธิ์เหนือ KeyPreview กล่าวคือเมื่อคุณกดปุ่มศรทั้งสี่ทิศ จะหมายถึงการเปลี่ยนโฟกัสไปยังคอนโทรลอื่นทันที แม้เขียนโค้ดดักไว้ในอีเวนต์ KeyDown ของฟอร์มก็ไม่มีความหมาย

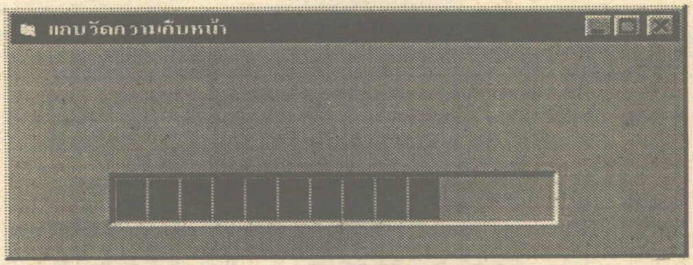
การใช้ Slider ตามศักยภาพทั้งหมดของมันนั้นค่อนข้างเป็นเรื่องซับซ้อน และบางทีก็นำไปประยุกต์ได้อย่างพิสดารแบบที่พวกเราคาดไม่ถึง จนกว่าจะพบจุดประสงค์การใช้งานเฉพาะหน้า สำหรับการแนะนำให้เห็นความแตกต่างระหว่างวิชวลเบสิก 3.0 กับ 4.0 นี้ ผมขอกล่าวถึงการใช้งานเพียงขั้นพื้นฐานให้คุณเห็นแนวว่ามันมีลักษณะการประยุกต์ใช้อย่างไรเฉพาะที่เห็นไปแล้ว

## ProgressBar

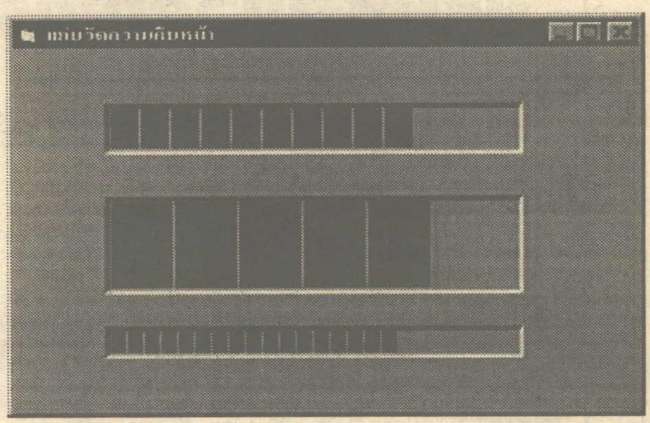
คอนโทรลตัวนี้ทำหน้าที่แสดงแถบวัดความคืบหน้าของผลการดำเนินการอะไรอย่างหนึ่ง คุณควรคิดเทียบกับคอนโทรล Gauge เพราะทำหน้าที่คล้ายกันในแง่การแสดงผลความคืบหน้าด้วยลักษณะแถบวัด แต่ต่างกันตรงที่ ProgressBar จะไม่มีแถบวัดประเภทที่คืบไปอย่างต่อเนื่อง แถบจะเรียงกันราวกับใครเอาอิฐมาวางต่อกันทีละก้อนจนกว่าจะเต็มพื้นที่แถบยาวตามแนวนอน

และอันเนื่องจากวิธีแสดงผลความคืบหน้าทำด้วยการแบ่งแถบเป็นก้อน ๆ ตอนผู้ใช้เขาดูก็จะเห็นค่าเพียงประมาณ แล้วตอนที่คุณออกแบบก็จะพบว่ามีพรอพเพอร์ตี้ส์หลายตัวที่มีผลกระทบกับจำนวนก้อนของแถบวัดไปด้วย ซึ่งก็จะส่งผลกับผู้ใช้อีกต่อในแง่ที่ช่วยให้เขาเห็นความคืบหน้าได้ละเอียดแค่ไหน ลองมาสร้าง ProgressBar ในรูปที่ 10 ขึ้นใหม่อีกสักสองรูปเพื่อเปรียบเทียบให้คุณเห็นข้อแตกต่างในการให้ความละเอียดอย่างชัดเจนตามรูปที่ 11

แถบวัดที่คุณเห็นทั้งสามขนาดนั้นความจริงผมกำหนดค่าความคืบหน้าให้กับพวกมันเท่ากันคือ 70 เปอร์เซ็นต์ ซึ่งเห็นด้วยสายตาแล้วออกจะต่างกันเอาเรื่องอยู่ สำหรับแถบแรกนั้นก็พอทำเนาล่ะครับ พอดูรู้อยู่ว่ามันเป็น 70 เปอร์เซ็นต์ (ถึงแม้รู้สึกว่าน่าจะเป็น 70 เปอร์เซ็นต์กว่า ๆ) แต่แถบที่สองนี่ใครบ้างคิดว่าต่ำกว่า 80 เปอร์เซ็นต์? ต้องคิดว่าเกิน 80 เปอร์เซ็นต์ขึ้นไปด้วยกันทั้งนั้น ส่วนแถบสุดท้ายนี่เข้าท่าหน่อย ดูปุ๊บรู้ทันทีว่าความคืบหน้าเข้ามาได้ประมาณ 3 ใน 4 ของการดำเนินการทั้งหมด ไม่เห็นเป็นอื่น ลองจินตนาการดูตอนนี้ คิดว่าผมกำลังฉายความคืบหน้าของแถบต่อเนื่องแต่ละแถบ คุณจะพบว่าสำหรับแถบที่สองนั้น กว่าจะเห็นว่ามันคืบหน้าแต่ละทีต้องรอนานเกินความเป็นจริงไปโข ส่วนแถบสุดท้ายจะแสดงความคืบหน้าได้ใกล้เคียงของจริงที่สุด




**รูปที่ 10 แถบวัดความคืบหน้าการดำเนินการที่กำลังแสดงว่าดำเนินการมาได้ประมาณ 70 เปอร์เซ็นต์แล้ว**




**รูปที่ 11 ความละเอียดที่ต่างกันของ ProgressBar**

อย่างไรก็ตาม แม้เห็นกันอยู่ว่า ProgressBar แสดงผลความคืบหน้าได้ไม่แน่นอนเท่ากับ Gauge แต่คุณสังเกตไหมครับว่าระยะหลัง ๆ นี่นักพัฒนาแอพพลิเคชันส่วนใหญ่หันมานิยมใช้รูปแบบแสดงผลความคืบหน้าแบบนี้กันเยอะมาก ลองตรองดูว่ามันเป็นเพราะอะไร ผมลองมองด้วยความรู้สึกและสายตาของผู้ใช้ พบว่าการแสดงความคืบหน้าเป็นก้อน ๆ มีผลทางการรับรู้ดีกว่าการแสดงแบบต่อเนื่องจริง ๆ เพราะการแสดงแบบเป็นก้อนนั้น ทำให้เรารู้สึกว่ากระบวนการดำเนินงานทั้งหมดถูกแบ่งเป็นขั้นเป็นตอนชัดเจน แล้วนำมาแสดงผลว่าล่วงเข้ามาได้กี่ก้าวแล้ว นี่คือความช่างสังเกตของนักออกแบบ ซึ่งแสดงให้พวกเราเห็นว่างานพัฒนาแอพพลิเคชันนั้น ความคิดสร้างสรรค์มีส่วนเกี่ยวข้องกับการรับรู้ของผู้ใช้เป็นอย่างมาก

เอาล่ะครับมาดูพรอพเพอร์ตี้ส์ที่สำคัญของ ProgressBar กัน

## พรอพเพอร์ตี้ส์หลัก

1. Min: เป็นค่าต่ำสุด ณ จุดเริ่มต้น โดยดีฟอลต์จะเป็น 0
2. Max: เป็นค่าสูงสุด ณ จุดปลาย โดยดีฟอลต์จะเป็น 100
3. Value: ค่าของตำแหน่งปัจจุบัน
4. Height: เป็นค่าส่วนสูงของคอนโทรล ซึ่งก็เหมือนกับส่วนสูงของคอนโทรลอื่น ๆ นั่นแหละครับ ต่างแต่ว่าส่วนสูงของคอนโทรลเช่น ProgressBar นี้ จะมีผลให้จำนวนของก้อนมันต่างกัน อย่างเช่นในรูปตัวอย่างนั้น สำหรับแถบที่สองผมเพียงขยายส่วนสูงของแถบแรกขึ้นไปสองเท่า และแถบสุดท้ายลดส่วนสูงของแถบแรกลงมาสองเท่า ผลก็ออกมาตามที่คุณเห็น ๆ กันแล้วอย่างชัดเจน นั่นคือยิ่งสูงจำนวนก้อนก็จะยิ่งน้อย ดูยากขึ้น
5. Width: เป็นค่าความกว้างของคอนโทรล อันนี้มีผลกับจำนวนก้อนอีกเหมือนกัน ยิ่งแคบจำนวนก้อนก็จะยิ่งน้อย ดูยากไม่ต่างกับเมื่อเพิ่มส่วนสูงให้คอนโทรล
6. BorderStyle: กำหนดเส้นกรอบของคอนโทรล เป็นอีกปัจจัยหนึ่งที่ส่งให้เกิดจำนวนก้อนมากน้อยแค่ไหน แต่ผลกระทบของ BorderStyle มีไม่มากนัก ผมพบว่ามันจะกระทบมากก็ตอนที่คุณกำหนด Height ไว้ต่ำ หรือกำหนด Width ไว้สูง หากคุณกำหนด BorderStyle เป็น Fixed Single (ค่าตัวเลขเป็น 1) จำนวนก้อนจะน้อยลงอย่างเห็นได้ชัด

คงต้องขอติดไว้คราวหน้าอีกสักสองสามตอนนะครับ สำหรับการยกเครื่องใหม่ของวิชวลเบสิกจากเวอร์ชัน 3.0 มาเป็น 4.0 นั้น มีเรื่องสำคัญมากมายให้พูดถึง ซึ่งผมก็จะคัดมาเสนอตามลำดับ สำหรับคราวหน้าก็ยังคงเป็นเรื่องเกี่ยวกับคัสตอมคอนโทรลใหม่ ๆ อยู่ ฉบับนี้ขอลาก่อนครับ สวัสดี

โปรดทราบ!
เพื่อเพิ่มความสะดวกในการติดต่องาน
บริษัท ซีเอ็ดยูเคชั่น (มหาชน) จำกัด
ได้เพิ่มเบอร์โทรศัพท์
อีก 30 หมายเลข จากเดิม
642-9800 30 สาย